

CERTIFIED COPY OF
PRIORITY DOCUMENT日本国特許庁
PATENT OFFICE
JAPANESE GOVERNMENTJC571 U.S. PTO
09/541980
04/03/00


別紙添付の書類に記載されている事項は下記の出願書類に記載されて
いる事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed
with this Office.

出願年月日
Date of Application:

1999年 4月 6日

出願番号
Application Number:

平成11年特許願第099406号

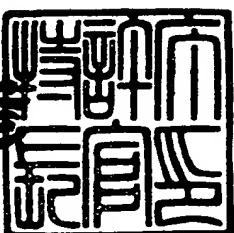
願人
Applicant(s):

ソニー株式会社

2000年 3月 3日

特許庁長官
Commissioner,
Patent Office

近藤 隆彦



【書類名】 特許願
【整理番号】 9801144303
【提出日】 平成11年 4月 6日
【あて先】 特許庁長官殿
【国際特許分類】 H04L 12/56
【発明者】
【住所又は居所】 東京都品川区北品川6丁目7番35号 ソニー株式会社
内
【氏名】 竹村 真一
【特許出願人】
【識別番号】 000002185
【氏名又は名称】 ソニー株式会社
【代表者】 出井 伸之
【代理人】
【識別番号】 100067736
【弁理士】
【氏名又は名称】 小池 晃
【選任した代理人】
【識別番号】 100086335
【弁理士】
【氏名又は名称】 田村 篤一
【選任した代理人】
【識別番号】 100096677
【弁理士】
【氏名又は名称】 伊賀 誠司
【手数料の表示】
【予納台帳番号】 019530
【納付金額】 21,000円

【提出物件の目録】

【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【包括委任状番号】 9707387

【プルーフの要否】 要

【書類名】 明細書

【発明の名称】 ネットワークシステム及びネットワーク制御方法、信号送受信装置

【特許請求の範囲】

【請求項1】 送り先を指定しないメッセージ及び特定のプロセス部を指定したメッセージを相互に送受信し得、第1状態と第2状態の何れにも自己の状態を変化し得るプロセス部を接続してなるネットワークシステムであって、

上記第1状態のプロセス部を記憶している上記第2状態のプロセス部を記憶する第1状態のプロセス部と、

上記第1状態の唯一のプロセス部を記憶する上記第2状態のプロセス部とを有し、

上記第1状態のプロセス部の数は1つであることを特徴とするネットワークシステム。

【請求項2】 上記第2状態の一のプロセス部は自己に関する情報を上記第1状態のプロセス部に複写し、上記第2状態の他のプロセス部は上記第1状態のプロセス部から上記一のプロセス部に関する情報を読み出すことにより、上記第2状態の各プロセス部は互いに各プロセス部に関する情報を共有することを特徴とする請求項1記載のネットワークシステム。

【請求項3】 プロセス部にアクセスするための情報を記述した情報を、上記第1状態のプロセス部に複写することにより、上記アクセスするための情報を記述した情報を各プロセス部間で共有することを特徴とする請求項2記載のネットワークシステム。

【請求項4】 上記第1状態のプロセス部及び当該第1状態の唯一のプロセス部のみを記憶する第2状態のプロセス部からなるグループ内では、上記送り先を指定しないメッセージ及び上記特定のプロセス部を指定したメッセージを相互に送受信し得、

異なるグループのプロセス部間では上記特定のプロセス部を指定したメッセージのみ送受信し得ることを特徴とする請求項1記載のネットワークシステム。

【請求項5】 一のグループの上記第1状態のプロセス部と、他のグループの第1状態のプロセス部との間でメッセージを交換することにより、両グループ間で唯一の第1状態のプロセスを決定する

ことを特徴とする請求項4記載のネットワークシステム。

【請求項6】 上記プロセス部は、通信エラーを検出するエラー検出手段を有する

ことを特徴とする請求項1記載のネットワークシステム。

【請求項7】 第1状態のプロセス部は、第2状態のプロセス部との間で通信エラーを検出した時、当該第2状態のプロセス部を記憶から削除する

ことを特徴とする請求項6記載のネットワークシステム。

【請求項8】 第2状態のプロセス部は、第1状態のプロセス部との間で通信エラーを検出した時、自己の状態を第1状態に変化させる

ことを特徴とする請求項6記載のネットワークシステム。

【請求項9】 上記プロセス部は、時間の経過を検知する時間経過検知手段を有する

ことを特徴とする請求項1記載のネットワークシステム。

【請求項10】 第1状態のプロセス部は、第2状態のプロセス部との間の通信が所定時間以上無かったことを検知した時、当該第2状態のプロセス部を記憶から削除する

ことを特徴とする請求項9記載のネットワークシステム。

【請求項11】 送り先を指定しないメッセージと特定のプロセス部を指定したメッセージとを相互に送受信し得、第1状態と第2状態の何れにも自己の状態を変化し得るプロセス部を接続してなるネットワークを、制御するネットワーク制御方法であって、

第1状態のプロセス部は、当該第1状態のプロセス部を記憶している上記第2状態のプロセス部を記憶し、

第2状態のプロセス部は、上記第1状態の唯一のプロセス部を記憶し、

上記第1状態のプロセス部の数を当該ネットワーク内で1つとする

ことを特徴とするネットワーク制御方法。

【請求項12】 上記第2状態の一のプロセス部に関する情報を上記第1状態のプロセス部に複写し、

上記第1状態のプロセス部から上記第2状態の他のプロセス部が上記第2状態の一のプロセス部に関する情報を読み出すことにより、

上記第2状態の各プロセス部は互いに各プロセス部に関する情報を共有することを特徴とする請求項11記載のネットワーク制御方法。

【請求項13】 プロセス部にアクセスするための情報を記述した情報を、上記第1状態のプロセス部に複写することにより、

上記アクセスするための情報を記述した情報を各プロセス部間で共有することを特徴とする請求項12記載のネットワーク制御方法。

【請求項14】 上記第1状態のプロセス部及び当該第1状態の唯一のプロセス部のみを記憶する第2状態のプロセス部からなるグループ内では、上記送り先を指定しないメッセージ及び上記特定のプロセス部を指定したメッセージを相互に送受信し得、

異なるグループのプロセス部間では上記特定のプロセス部を指定したメッセージのみ送受信し得る

ことを特徴とする請求項11記載のネットワーク制御方法。

【請求項15】 一のグループの上記第1状態のプロセス部と、他のグループの第1状態のプロセス部との間でメッセージを交換することにより、両グループ間で唯一の第1状態のプロセスを決定する

ことを特徴とする請求項14記載のネットワーク制御方法。

【請求項16】 第1状態のプロセス部は、第2状態のプロセス部との間で通信エラーを検出した時、当該第2状態のプロセス部を記憶から削除する

ことを特徴とする請求項11記載のネットワーク制御方法。

【請求項17】 第2状態のプロセス部は、第1状態のプロセス部との間で通信エラーを検出した時、自己の状態を第1状態に変化させる

ことを特徴とする請求項11記載のネットワーク制御方法。

【請求項18】 第1状態のプロセス部は、第2状態のプロセス部との間の通信が所定時間以上無かったことを検知した時、当該第2状態のプロセス部を記憶

から削除する

ことを特徴とする請求項11記載のネットワーク制御方法。

【請求項19】 特定の宛先を指定したメッセージ及び送り先を指定しないメッセージを少なくとも生成可能なメッセージ生成手段と、

送信されてきたメッセージを受信して内容解析を行うメッセージ解析手段と、

ネットワーク上に接続された他の機器の有無及び他の機器が第1状態と第2状態の何れの状態であるかに応じて、自己の状態を第1、第2状態の何れかに変化させる状態制御手段と、

自己及び他の機器に関する情報を記憶可能な記憶手段とを有し、

自己に関する情報を記憶している第1状態の唯一の他の機器がネットワークに接続されている時には、自己の状態を第2状態として上記第1状態の他の機器を記憶し、

第1状態の自己を記憶している第2状態の他の機器がネットワークに接続されている時には、上記自己を記憶している第2状態の他の機器に関する情報を記憶する

ことを特徴とする信号送受信装置。

【請求項20】 自己に関する情報を記憶している第1状態の唯一の他の機器がネットワークに接続されている時には、自己に関する情報を上記第1状態の他の機器に複写し、必要に応じて当該第1状態の機器が記憶する他の第2状態の機器に関する情報を読み出す

ことを特徴とする請求項19記載の信号送受信装置。

【請求項21】 ネットワークに接続されている他の機器にアクセスするための情報を記述した情報を、上記第1状態の他の機器に複写し、必要に応じて当該第1状態の機器が記憶する上記アクセスするための情報を記述した情報を読み出す

ことを特徴とする請求項20記載の信号送受信装置。

【請求項22】 上記第1状態の機器及び当該第1状態の唯一の機器のみを記憶する第2状態の機器からなるグループ内に接続されている時には、上記送り先を指定しないメッセージ及び上記特定の宛先を指定したメッセージを相互に送受

信し得、異なるグループの機器に対しては上記特定の宛先を指定したメッセージのみ送受信し得る

ことを特徴とする請求項21記載の信号送受信装置。

【請求項23】自己が第1状態であるとき、他のグループの第1状態の機器との間でメッセージを交換することにより、両グループ間で唯一の第1状態の機器を決定する

ことを特徴とする請求項22記載の信号送受信装置。

【請求項24】自己が第2状態であるとき、他のグループの第1状態の機器からのメッセージを自己が属するグループ内の第1状態の機器に転送することを特徴とする請求項22記載の信号送受信装置。

【請求項25】通信エラーを検出するエラー検出手段を有する

ことを特徴とする請求項19記載の信号送受信装置。

【請求項26】自己が第1状態であるとき、第2状態の機器との間で通信エラーを検出したならば、当該第2状態の機器を記憶から削除することを特徴とする請求項25記載の信号送受信装置。

【請求項27】自己が第2状態であるとき、第1状態の機器との間で通信エラーを検出したならば、自己の状態を第1状態に変化させることを特徴とする請求項25記載の信号送受信装置。

【請求項28】時間の経過を検知する時間経過検知手段を有する

ことを特徴とする請求項19記載の信号送受信装置。

【請求項29】自己が第1状態であるとき、第2状態の機器との間で所定時間以上の無通信となったことを検知したならば、当該第2状態の機器を記憶から削除することを特徴とする請求項19記載の信号送受信装置。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

本発明は、例えば複数のノードのアプリケーションプログラム或いはオブジェクト間でメッセージや各種情報を伝送するネットワークシステム及びそのネット

ワークシステムを制御するネットワーク制御方法、当該ネットワークシステムに接続される信号送受信装置に関する。

【0002】

【従来の技術】

従来より、複数のノードのアプリケーションプログラム或いはオブジェクト（以下、これら全てを含む上位概念をプロセスと呼ぶこととする。）間でメッセージや各種情報を伝送するネットワークシステムにおいて、例えばサーバとなるべきプロセスを決定する場合、サーバとなり得る機能を備えたプロセスは、ネットワーク内で宛先を指定しない単純なブロードキャストを用いて、現時点でネットワーク内に存在するサーバを捜し、当該サーバが発見できなければ自らがサーバになるという方により、サーバ（すなわちサーバプロセス）が決定されている。

【0003】

したがって、このネットワーク内のサーバ以外のプロセス（以下、クライアントプロセスと呼ぶ）が何らかの処理を行う際には、先ず、上記ブロードキャストを用いてネットワーク上のサーバプロセスを発見し、当該クライアントプロセスからサーバプロセスに対して要求を行い、所望の処理を実行する。

【0004】

また、上述のような単純なブロードキャストによるサーバプロセスの決定処理は、例えばネットワーク上の各ノードにおいて電源を投入した時や、各ノードがネットワークに新たに接続された時、或いはネットワーク上の各ノードが何らかの処理命令を入力する時などに、行われることが多い。

【0005】

【発明が解決しようとする課題】

上述したように、従来のネットワークシステムにおいては、上記宛先を指定しない単純なブロードキャストを用いてサーバプロセスを決定するようになされている。このため、例えば数万或いはそれ以上の膨大な数のプロセスが接続された大規模なネットワーク上で、当該単純なブロードキャストによりサーバプロセスを決定するような手法を採用することは、当該サーバプロセス決定に長時間を要することから、非現実的である。言い換えれば、上記単純なブロードキャスト

によるサーバプロセスの決定処理は、検索範囲が限定された小規模なネットワークにのみ適用可能である。

【0006】

また、上述したように、従来のネットワークシステムにおいては、例えば電源投入時やネットワークへの接続時或いは処理命令の入力時等に、上記単純なプロードキャストによるサーバプロセスの決定処理が行われている。このため、例えばサーバプロセスに何らかの障害が発生した場合に、各クライアントプロセスにおいて当該サーバプロセスに障害が発生したことを知り得るのは、上記電源投入時や接続時、処理命令の入力時等のサーバプロセス決定処理時に限られ、したがって、上記サーバプロセスの障害発生を早期に発見することが出来ず、当該障害を復旧するまでの時間や処理の開始が遅れてしまう。言い換れば、サーバプロセスの障害復旧がなされるまで、各クライアントプロセスは、ネットワーク上の処理に制限を受けてしまう。

【0007】

そこで、本発明はこのような状況に鑑みてなされたものであり、様々なネットワーク構成に柔軟に対応でき、尚かつネットワーク上で障害が発生しても、各プロセスが所望の処理を継続可能とする、ネットワークシステム及びネットワーク制御方法、信号送受信装置を提供することを目的とする。

【0008】

【課題を解決するための手段】

本発明のネットワークシステムは、送り先を指定しないメッセージ及び特定のプロセス部を指定したメッセージを相互に送受信し得、第1状態と第2状態の何れにも自己の状態を変化し得るプロセス部を接続してなるネットワークシステムであって、上記第1状態のプロセス部を記憶している上記第2状態のプロセス部を記憶する第1状態のプロセス部と、上記第1状態の唯一のプロセス部を記憶する上記第2状態のプロセス部とを有し、上記第1状態のプロセス部の数は1つであることにより、上述した課題を解決する。

【0009】

本発明のネットワーク制御方法は、送り先を指定しないメッセージと特定のプ

口セス部を指定したメッセージとを相互に送受信し得、第1状態と第2状態の何れにも自己の状態を変化し得るプロセス部を接続してなるネットワークを、制御するネットワーク制御方法であって、第1状態のプロセス部は、当該第1状態のプロセス部を記憶している上記第2状態のプロセス部を記憶し、第2状態のプロセス部は、上記第1状態の唯一のプロセス部を記憶し、上記第1状態のプロセス部の数を当該ネットワーク内で1つとすることにより、上述した課題を解決する。

【0010】

本発明の信号送受信装置は、特定の宛先を指定したメッセージ及び送り先を指定しないメッセージを少なくとも生成可能なメッセージ生成手段と、メッセージを受信して内容解析を行うメッセージ解析手段と、ネットワーク上に接続された他の機器の有無及び他の機器が第1状態と第2状態の何れの状態であるかに応じて自己の状態を第1、第2状態の何れかに変化させる状態制御手段と、自己及び他の機器に関する情報を記憶可能な記憶手段とを有し、自己に関する情報を記憶している第1状態の唯一の他の機器がネットワークに接続されている時には自己の状態を第2状態として上記第1状態の他の機器を記憶し、第1状態の自己を記憶している第2状態の他の機器がネットワークに接続されている時には上記自己を記憶している第2状態の他の機器に関する情報を記憶することにより、上述した課題を解決する。

【0011】

【発明の実施の形態】

本発明の好ましい実施の形態について、図面を参照しながら説明する。

【0012】

図1には、本発明のネットワークシステム及びネットワーク制御方法、信号送受信装置が適用される一実施の形態としてのネットワークシステムの概略構成を示す。

【0013】

この図1において、本発明実施の形態のネットワークシステムは、何らかの情報やサービス提供等の様々な処理を行い得るプロセスP1、P2、P3、P4、

P5と、これら各プロセスP1, P2, P3, P4, P5を備えるノード（例えば信号送受信機能を備えたコンピュータ等）A1, A2, A3, A4, A5と、バス型のネットワークセグメントB1及びB2と1対1接続のセグメントB3とからなり、これらによりメッセージを伝達可能なネットワークが構成されている。

【0014】

すなわち、この図1に示すネットワークシステムでは、バス型のセグメントB1に接続されたプロセスP1, P2, P3（ノードA1, A2, A3）から成るネットワークN1と、バス型のセグメントB2に接続されたプロセスP3, P4（ノードA3, A4）から成るネットワークN2と、1対1接続のセグメントB3に接続されたプロセスP4, P5（ノードA4, A5）から成るネットワークN3とが、結合して一つのネットワークを構成している例を挙げている。

【0015】

ここで、当該ネットワーク内の各プロセスは、他の全てのプロセスとの間でメッセージを送受信できるように、メッセージが正しくルーティングされていることとする。

【0016】

また、各プロセスは、それぞれが自己のアドレスや機能、仕様、動作状況等（以下適宜、機能等と呼ぶ。）についての情報を、ネットワーク内の他のプロセスに対してメッセージとして送信可能となされている。他のプロセスから上記機能等の情報をメッセージとして受信した各プロセスは、それら他のプロセスからの機能等の情報を記憶することが可能である。さらに、各プロセスは、自己のプロセスの上記機能等が変化した場合、その変化を知らせるためのメッセージを、他のプロセスに対して送信可能である。他のプロセスから上記機能等が変化したことを知らせるメッセージを受信した各プロセスは、そのメッセージに応じて、既に記憶している他のプロセスの機能等の情報を変更（更新）することになる。なお、各プロセスは、それぞれ他のプロセスのアドレスや機能等の情報を複写、転送することも可能である。また、各プロセスは、何らかの障害によって存在が確認できないプロセスがある場合、そのプロセスのアドレスや機能等の情報を記憶

から削除することができる。

【0017】

当該ネットワークシステム上の各プロセスは、それぞれが、ネットワーク内の他の全てのプロセスのアドレスや機能等についての情報を管理するプロセスと成り得るものであり、また、各プロセスはネットワーク内の他のプロセスのアドレスや機能等の情報を他のプロセスに委ねるプロセスと成り得るものである。以下の説明では、ネットワーク内の他の全てのプロセスのアドレスや機能等の情報を管理することになったプロセスを親プロセスと呼び、当該親プロセスによりアドレスや機能等の情報を管理されることになったネットワーク内の他のプロセスを子プロセスと呼ぶこととする。親プロセスとしての状態、子プロセスとしての状態は、他のプロセスとの間のメッセージ交換やその他の要因によって変化するものである。なお、初期状態では、全てのプロセスは親プロセスとなる。

【0018】

また、親プロセスは自己が親プロセスであることを記憶し、子プロセスは自己が子プロセスであることを記憶する。子プロセスは一つの親プロセスを記憶し、親プロセスは自己を親プロセスとして記憶している全ての子プロセスを記憶する。図1の例を用いて説明すると、例えばバス型のセグメントB1に接続されたプロセスP1, P2, P3から成るネットワークN1のみが形成され、例えばプロセスP1が親プロセス、プロセスP2及びP3がそれぞれ子プロセスであるとした場合、プロセスP1は、自己が親プロセスであることを記憶し、且つプロセスP2, P3が子プロセスであることを記憶する。プロセスP2, P3は、それぞれ自己が子プロセスであることを記憶し、且つ親プロセスがプロセスP1であることを記憶する。

【0019】

さらに、同じ親プロセスを記憶する複数のプロセス（子プロセスと親プロセスの両方を含む）は、1つのグループを形成する。図1の例を用いて説明すると、例えばバス型のセグメントB1及びB2に接続されたプロセスP1, P2, P3, P4から成るネットワークN1とN2が結合したネットワークのみが形成され

、例えばプロセスP1が親プロセスとし、プロセスP2, P3, P4がそれぞれ子プロセスであるとした場合、これらプロセスP1, P2, P3, P4で一つのグループが形成される。なお、初期状態では、上述したように全てのプロセスが親プロセスであるため、当該初期状態のときの全てのプロセスは個々に独自のグループを形成している。

【0020】

上述した同一グループに属する各子プロセスは、自己のアドレスや機能や仕様、動作状況等の情報を同グループの親プロセスに複写する。したがって、親プロセスは、当該グループ内の全ての子プロセスのアドレスや機能等の情報を保持することが可能となる。さらに、親プロセスは、自己が記憶している全ての子プロセスに対して、その記憶している各アドレスや機能等の情報を複写することが可能となっている。一方、同一グループ内の子プロセスは、同グループの親プロセスにアクセスして、当該親プロセスが保持する全ての情報を取得することが可能となっている。これらのことから、同一グループ内の各プロセスが保持する種々の情報は、当該グループ内の各プロセスにおいて共有され、また、同一グループ内の全てのプロセスは、同グループ内の各プロセスの全ての情報を持つことが可能になる。さらに、各プロセスは、自らが提供するサービスにアクセスするための手段を記述した情報を持ち、当該アクセスのための情報も、前述同様に複写によって全てのプロセスが持つことができる。したがって、同一グループ内の全てのプロセスは、同グループ内の他のプロセスのサービスにアクセス可能である。以上により、各種の障害が発生したり、ネットワークの構成が変化するようなことがあったとしても、同一グループ内の各プロセスは、各種の情報を共有でき、また、同グループ内の他のプロセスへのアクセスが可能となっている。

【0021】

さらに、同一グループ内の各プロセス間では、宛先を指定しないブロードキャストによるメッセージの送信が可能であるが、例えば未だグループに属していないプロセスが当該グループに新たに接続等された場合や、同一グループに属しないネットワーク上のプロセスに対しては、ブロードキャストによるメッセージの送信は許可されず、アドレスを指定したメッセージの交換のみが可能となされる

。図1の例を用いて説明すると、例えばプロセスP1, P2, P3, P4にて一つのグループが形成されている場合において、例えばプロセスP5がプロセスP4に新たに接続等され、未だ当該グループに属していないような場合、例えば各プロセスP1, P2, P3, P4間では相互にブロードキャストによるメッセージの送信が可能であるが、各プロセスP1, P2, P3, P4とプロセスP5との間ではアドレスを指定したメッセージの交換のみが行われる。

【0022】

また、例えばあるグループに新たに接続等されて未だ当該グループに属さないプロセスがある場合、当該グループに既に属している全てのプロセスは、上記新たに接続等されて未だ当該グループに属していないプロセスに対して、このグループの親プロセスを知らせるメッセージを送信可能となされている。

【0023】

ここで、例えばある別個の2つのグループに属するプロセス間が接続された場合、当該接続された各プロセスは、それぞれが属するグループの親プロセスを知らせるメッセージを、接続相手方のグループのプロセスに送る。当該接続相手方のグループのプロセスからそのグループの親プロセスを知らせるメッセージを受け取ったプロセスは、自己が属しているグループの親プロセスに対してそのメッセージを転送する。図1の例を用いて説明すると、例えばプロセスP1が親プロセスでプロセスP2, P3が子プロセスとなっている一つのグループと、プロセスP5が親プロセスでプロセスP4が子プロセスとなっている他の一つのグループとがあるような場合において、例えばプロセスP3とP4の間が新たに接続されたとすると、上記プロセスP3は自己が属するグループの親プロセスP1を知らせるメッセージをプロセスP4に送り、また、このプロセスP4は自己が属するグループの親プロセスP5を知らせるメッセージをプロセスP3に送る。プロセスP4からそのグループの親プロセスP5を知らせるメッセージを受け取ったプロセスP3は、そのメッセージを自己が属するグループの親プロセスP1に転送する。同様に、プロセスP3からそのグループの親プロセスP1を知らせるメッセージを受け取ったプロセスP4は、そのメッセージを自己が属するグループの親プロセスP5に転送する。

【0024】

上述のようにして互いに他のグループの親プロセスの存在を知らせるメッセージを受け取った各親プロセスは、それぞれ相手方の親プロセスに対してメッセージを交換し、何れか一方が子プロセスの状態に変化する。図1の例を用いて説明すると、例えばプロセスP1が一方のグループの親プロセスで、プロセスP5が他方のグループの親プロセスであるような場合、これらプロセスP1とP5は、互いにメッセージを交換して、何れか一方が子プロセスに変化する。ここで、何れの親プロセスが子プロセスに変化するかの判断基準としては種々考えられるが、その一例として、自己のグループ内に所属する子プロセスの数が少ない方の親プロセスが子プロセスに変化する例や、互いの機能を比較して劣っている方が子プロセスに変化する例等が考えられる。

【0025】

上記別個の2つのグループの接続により、子プロセスの状態に変化することになった親プロセスは、自己が現時点で記憶している全子プロセスに対して新たな親プロセスを知らせ、また、自己が現時点で記憶している全子プロセスに関する情報を新たな親プロセスに転送する。図1の例を用いて説明すると、例えばプロセスP1が一方のグループの親プロセスで、プロセスP5が他方のグループの親プロセスであったような場合において、プロセスP5が子プロセスに変化したとすると、当該プロセスP5はその子プロセスP4に対して、新しい親プロセスがプロセスP1であるというメッセージを送り、また、プロセスP1に対しては自己のアドレスや機能等とプロセスP4のアドレスや機能等の各種情報をメッセージとして送る。

【0026】

上述したように自己が属するグループの親プロセスが子プロセスに変化するのに伴って、当該親プロセスから新たな親プロセスを示すメッセージを受け取った子プロセスは、その新たな親プロセスを記憶し、さらに当該新たな親プロセスに対して自己の機能等の情報をメッセージとして送信する。これにより、当該子プロセスは、上記新たな親プロセスに記憶されることになる。図1の例を用いて説明すると、例えばプロセスP5が親プロセスでプロセスP4が子プロセスである

場合において、プロセスP5から新たな親プロセスがプロセスP1であることを知らされたプロセスP4は、当該新たな親プロセスP1を記憶し、また、この新たな親プロセスP1に対して自己の機能等をメッセージとして送る。これにより、親プロセスP1は、プロセスP4を子プロセスとして記憶する。

【0027】

以上のようにして、例えば別個の2つのグループが接続され、一方のグループの親プロセスが子プロセスに変化し、残った方の親プロセスが自己及び他のグループの各プロセスを子プロセスとして記憶することにより、これら2つのグループは一つのグループに収束することになる。また、このように1つのグループに纏められることで、元々別々のグループに属していた各プロセス間では宛先を指定しないブロードキャストによるメッセージの送信が可能となる。

【0028】

次に、例えばグループ内で何らかの障害等が発生し、例えば親プロセスがそのグループ内のある子プロセスの存在を確認できなくなったとき、当該親プロセスは存在を確認出来なくなった子プロセスについての記憶を削除する。図1の例を用いて説明すると、例えばプロセスP1が親プロセスでプロセスP5が子プロセスとなっている場合において、例えば1対1接続のセグメントB3（ネットワークN3）に何らかの障害が発生したり、プロセスP5が当該ネットワークから外されたりなどして、親プロセスP1が子プロセスP5と通信不能になった場合、親プロセスP1は自己が記憶している子プロセスの中から上記子プロセスP5を削除する。

【0029】

また、例えばグループ内で何らかの障害等が発生し、例えばある子プロセスが自己が記憶している親プロセスの存在を確認できなくなった場合、当該子プロセスは所属していたグループから自己が離脱したと判断して、初期状態と同様に自らが独自のグループを形成し且つ親プロセスの状態に変化する。図1の例を用いて説明すると、例えばプロセスP1が親プロセスでプロセスP5が子プロセスであったような場合において、例えば1対1接続のセグメントB3（ネットワークN3）に何らかの障害が発生したり、プロセスP5が当該ネットワークから外さ

れたりなどして、当該プロセスP5が親プロセスP1と通信不能になった場合、このプロセスP5は独自のグループを形成し自ら親プロセスに変化する。

【0030】

上述したように、本実施の形態のネットワークシステムは、一つのグループ内では親プロセスが一つしか存在しない状態になるべく速やかに移行する。すなわち、本実施の形態のネットワークシステムにおいては、例えばグループ内のネットワークに何らかの障害が発生したり、逆に、障害が取り除かれてネットワークが復帰した場合、或いは、ネットワークの構成が変更されてネットワークの様相が変化した場合など、何れの場合も、新しい様相に適合し、速やかに一グループ内に親プロセスは一つとなる状態に移行することが可能である。また、このように、一グループ内に親プロセスが一つのみ存在する状態になったとき、当該グループ内の全ての子プロセスは一つの親プロセスを記憶し、親プロセスは全ての子プロセスを記憶していることになる。すなわち、図1の例を用いて説明すると、例えばプロセスP1が親プロセスで他の各プロセスP2, P3, P4, P5が子プロセスである場合において、親プロセスP1は他の全ての子プロセスP2, P3, P4, P5を記憶し、各子プロセスP2, P3, P4, P5は親プロセスとしてプロセスP1を記憶している。

【0031】

以上、本発明実施の形態のネットワークシステムの基本動作について概略的に説明したが、これらの動作は図2～図7のフローチャートにて表すことができる。

【0032】

図2には、上述した本発明実施の形態のネットワークシステムの基本動作のうち、例えば、あるプロセスがネットワークに接続された時、或いはあるプロセスに電源が投入された時に、当該プロセスが親プロセス又は子プロセスの何れかの状態になるまでの流れを示している。

【0033】

この図2において、先ずステップS1の処理として、あるプロセスがネットワークに接続或いは電源投入されると、次のステップS2に処理が進み、このステ

ップS2にて当該ネットワーク内に既に親プロセスが存在するか否かが判断される。このステップS2の判断は、グループ内に自己以外のプロセスが存在するか否か、或いは、グループ内の他のプロセスから親プロセスの存在を知らせるメッセージが送られてきたか等により行われる。

【0034】

このステップS2において、他に親プロセスが存在すると判断した場合、ステップS3にて自プロセスが子プロセスであると認識し、次のステップS4にて自プロセスの機能等の情報を親プロセスにメッセージとして送信する。

【0035】

一方、ステップS2において、他に親プロセスが存在しないと判断した場合、ステップS5にて自己が親プロセスであることをネットワーク内の他のプロセスにブロードキャストにより送信する。

【0036】

その後、ステップS6にて、ネットワーク内の他の子プロセスの機能等の情報のメッセージを各子プロセスから受信し、さらにステップS7にて、このネットワーク内の各プロセスとの間でグループを構成する。

【0037】

図3には、本発明実施の形態のネットワークシステムの基本動作のうち、例えば、ある別個の2つのグループに属するプロセス間が接続された場合に、当該接続されたプロセスが行う処理の流れを示す。

【0038】

この図3において、ある別個の2つのグループに属するプロセス間が接続された場合、当該接続されたプロセスは、ステップS11として、他のグループのプロセスから当該他のグループの親プロセスを知らせるメッセージを受け取ることになる。

【0039】

このステップS11のように、他のグループの親プロセスを知らせるメッセージを受け取ったときのプロセスは、ステップS12のように、自グループ内の親プロセスに対して、上記他のグループの親プロセスを知らせるメッセージを転送

する。

【0040】

図4には、本発明実施の形態のネットワークシステムの基本動作のうち、例えば、図3のように自グループ内の子プロセスから他グループの親プロセスを知らせるメッセージを受信した場合の親プロセスにおける処理の流れを示す。

【0041】

この図4において、ステップS21にて、自グループ内の子プロセスから他のグループの親プロセスを知らせるメッセージを受信すると、当該メッセージを受け取った親プロセスは、ステップS22として他グループの親プロセスとの間でメッセージを交換する。

【0042】

次に、ステップS22として、親プロセスは、他グループの親プロセスの自己の何れが親プロセスとなるべきかを判断する。この判断は前述したように、自己のグループ内の子プロセス数や機能の比較等により行うことができる。

【0043】

このステップS23にて自己が親プロセスになると判断した場合は、ステップS27以降の処理に進み、他グループの親プロセスを自己の親プロセスにすると判断した場合は、ステップS24以降の処理に進む。

【0044】

ステップS23にて他グループの親プロセスを自己の親プロセスにすると判断した場合、ステップS24にて自ら子プロセスに状態に変化し、ステップS25にて自己が記憶していた各子プロセスに対して新たな親プロセス（他グループの親プロセス）を知らせるメッセージを送信する。

【0045】

一方、ステップS23にて自己が親プロセスになると判断した場合、ステップS26にて、2つのグループを合わせた新たなグループの親プロセスとなる。次いで、当該親プロセスは、ステップS27として、新たにグループ内に加わった子プロセス（他のグループに属していた子プロセス）からメッセージを受信し、これら子プロセスの情報を記憶する。

【0046】

図5には、本発明実施の形態のネットワークシステムの基本動作のうち、例えば、上述のように2つのグループが接続されることで、自己が属していたグループの親プロセスが子プロセスに変化して新たな親プロセスが存在することになった場合の、子プロセスにおける処理の流れを示す。

【0047】

この図5において、ステップS41にて、自己が記憶していた親プロセスから、新しい親プロセスについてのメッセージを受信すると、当該メッセージを受け取ったプロセスは、ステップS42にて新たな親プロセスを記憶する。

【0048】

次に、このプロセスは、ステップS43において、新しい親プロセスに対して自プロセスの機能等の情報を送信し、自プロセスを子プロセスとして新しい親プロセスに記憶させる。

【0049】

図6には、本発明実施の形態のネットワークシステムの基本動作のうち、例えばグループ内で何らかの障害等が発生し、例えば親プロセスがそのグループ内のある子プロセスの存在を確認できなくなったときの、当該親プロセスにおける処理の流れを示す。

【0050】

この図6において、親プロセスは、ステップS51にて自己が属するグループ内の全ての子プロセスの存在を確認出来るか否かの判断を行う。ここで、グループ内の全ての子プロセスの存在を確認できた場合は処理を終了し、ある子プロセスの存在を確認できなかったときはステップS52の処理に進む。

【0051】

親プロセスは、ステップS51にて存在を確認できない子プロセスがあると判断した場合、ステップS52にて、その存在を確認できない子プロセスを記憶から削除する。

【0052】

図7には、本発明実施の形態のネットワークシステムの基本動作のうち、例え

ば、例えばグループ内で何らかの障害等が発生し、例えばある子プロセスが自己が記憶している親プロセスの存在を確認できなくなったときの、当該子プロセスにおける処理の流れを示す。

【0053】

この図7において、子プロセスは、ステップS61にて自己が属するグループ内の親プロセスの存在を確認出来るか否かの判断を行う。ここで、グループ内の親プロセスの存在を確認できた場合は処理を終了し、親プロセスの存在を確認できなかったときはステップS62の処理に進む。

【0054】

子プロセスは、ステップS61にて親プロセスの存在を確認できなかった場合、ステップS62にて、自らが独自のグループを形成して、そのグループの親プロセスに変化する。

【0055】

以上、図2から図7のフローチャートを用いて、本発明実施の形態のネットワークシステムの幾つかの基本動作を説明したが、本発明実施の形態のネットワークシステムの動作は、図8～図10に示すように、C言語に準じた手法で表現することもできる。なお、図8～図10は本来は1つの図上に表すべきものであるが、紙面の都合上、3つの図に分けて示している。

【0056】

図8は、プロセスが親プロセスである際に、受信したメッセージや発生したエラーに対して、他のプロセスにメッセージを送信したり、子プロセス又は中間状態に状態を変化させる処理の流れを表している。図9は、親プロセスが別のグループの親プロセスとメッセージを交換中の間状態である際に、受信したメッセージや発生したエラーに対して、他のプロセスにメッセージを送信したり、子プロセス又は親プロセスに状態を変化させる処理の流れを表している。図10は、プロセスが子プロセスである際に、受信したメッセージや発生したエラーに対して、親プロセスにメッセージを転送したり、他のプロセスにメッセージを送信したり、親プロセスに状態を変化させる処理の流れを表している。

【0057】

以下に、これら図8～図10において使用している各引数や操作、関数等について説明する。

【0058】

プロセスは、以下のメッセージを使用する。メッセージには、0個以上の引数が付帯する。

【0059】

`M_NOTIFY addr`は、自分の親プロセスのアドレスを一方的に他のプロセスに通知する際に用いる。この引数の`addr`は親のアドレスを表す。なお、この通知は、例えば所定時間毎（一定時間毎）に行われ、例えば複数のプロセスがある場合にはアドレスのリストに載っているプロセスにブロードキャストを用いて行われ、例えば1対1のケーブル接続の時にはその相手方のプロセスに対して行われる。

【0060】

`M_FORWARD addr`は、親プロセスが変更されたことを親プロセスから子プロセスへ知らせる際に用いる。この引数の`addr`は新しい親プロセスのアドレスを表す。

【0061】

`M_PREPORT addr`は、子プロセスから親プロセスに対して、他の親プロセスの存在を知らせる際に用いる。この引数の`addr`は他の親プロセスのアドレスを表す。

【0062】

`M_NEGOTIATE addr`は、親プロセス間で、新しい親プロセスを決定する際に用いる。なお、この因数の`addr`は、メッセージの前に付加されている送信元のアドレスと宛先アドレスによって知ることができる場合には省略可能である。

【0063】

`M_IAM addr`は、自分が新たな親プロセスであることを他の親プロセスに宣言する際に用いる。この引数の`addr`は新たな親プロセスのアドレスを表す。なお、この因数の`addr`は、メッセージの前に付加されている送信元のアドレスと宛先アドレスによって知れる場合には省略可能である。

【0064】

`M_YOUARE`は、自分が新たな子プロセスであることを他の親プロセスに宣言する

際に用いる。

【0065】

M_BUSYは、直前のメッセージの再送を要求する際に用いる。

【0066】

M_JOIN addrは、子プロセスとして親プロセスに登録する際に用いる。なお、この因数のaddrは、メッセージの前に付加されている送信元のアドレスと宛先アドレスによって知ることができる場合には省略可能である。

【0067】

また、上述したメッセージを受信する以外に、内部的に以下のイベントが発生する。

【0068】

E_ERRORは、障害が発生したことを示す。

【0069】

E_TIMEOUTは、特定のメッセージの受信又はイベントの発生より一定時間が経過したことを示す。

【0070】

さらに、システムは以下の何れかの状態を持つ。

【0071】

S_PARENTは、親プロセスである状態を示す。

【0072】

S_NEGOTIATINGは、親プロセスでも子プロセスでもない状態を示す。

【0073】

S_CHILDは、子プロセスである状態を示す。

【0074】

また、システムは以下の内部変数を持つ

Pは、親プロセスのアドレスを示す。

【0075】

C[]は、子プロセスのアドレス（複数）を示す。

【0076】

Mは、自プロセスのアドレス（定数）を示す。

【0077】

Sは、現在の状態を示す。

【0078】

Eは、最近受信したメッセージまたは最近発生したイベントを示す。

【0079】

また、以下の操作を定義する

`send<destination><message>....`は、<destination>に対してメッセージ<message>....を送信することを示す。<destination>が複数の場合にはそれぞれに対して送信する。

【0080】

`broadcast<message>....`は、メッセージ<message>....をブロードキャストすることを示す。

【0081】

`add_C<addr>`は、C[]に<addr>を加えることを示す。

【0082】

`empty_C`は、C[]を空にすることを示す。

【0083】

`assign_P<addr>`：Pの値を<addr>に置き換えることを示す。

【0084】

`trans<state>`は、状態Sを<state>に変更することを示す。

【0085】

`wait`は、メッセージを受信するか、イベントが発生するまで待つことを示す。

受信したメッセージ又は発生したイベントはEに格納される。

【0086】

`ignore`は、何もしないことを示す。

【0087】

さらに、以下の関数を定義する。

【0088】

`decide(<a>,)`は、`<a>`と``のどちらかを選んで返すことを示す。

【0089】

そのほか、本実施の形態のネットワークシステムにおいて、考え得る拡張例としては、以下のようなことを挙げることができる。

【0090】

すなわち例えば、親プロセスとの通信エラーを、いわゆる PING (IP の上位プロトコルの一種である ICMP を用いて相手先ホストに対して返答要求を送出するプログラム) と一定時間応答がない場合のタイムアウトによって検出する例や、親プロセスとの通信エラーを、TCP/IP (Transmission Control Protocol/Internet Protocol) などのコネクションの切断によって検出する例、キャリア (搬送波信号) を検知することによって親プロセスとの通信エラーを検出する例、子プロセスとの通信エラーを検出し、エラーとなった子プロセスは C[] から取り除く例、メッセージのソースアドレスをチェックして不正なものは無視する例、パスワード等によるチェック機構を用いて悪意のある攻撃を防ぐ例、メッセージにグループの識別子を付けてグループ毎に親プロセスを決定する例、一つのプロセスが複数のグループに参加する例、ブロードキャストに加えて特定のアドレスへの片方向の通知を使用する例、親プロセスと子プロセスの関係を階層構造にする例、信頼性のない通信 (例えばいわゆる UDP など) のために改良する例、受信した単位 (例えばパケット) の内容や送信したパケットの内容によってタイムアウトを変更する例などが考えられる。

【0091】

次に、プロセス間で情報を複写する場合の複写機構について以下に説明する。

【0092】

当該複写を行う場合、各プロセスは以下の形式のメッセージを使用する。

【0093】

すなわち、`M_COPY addr<情報>`と、`M_DELETE addr`の形式のメッセージを使用する。なお、このメッセージにおける`addr`は複写元のアドレスを示す。

【0094】

さらにこの場合のプロセスは、図11に示すような構成のテーブルを用いて情報を記憶する。

【0095】

この図11に示す構成のテーブルにおけるaddrは複写元のプロセスのアドレスを示す。D[]は情報を複写したプロセスの一覧であり、これにより情報が複写済みであるかどうかを判断するために使用する。

【0096】

なお、上記複写を行う場合の流れを、C言語に準じた手法で表現すると、図12のよう表すことができる。この図12は、子プロセスから親プロセスに情報を複写する際、及び親プロセスから全ての子プロセスに情報を複写する際の処理の流れを表している。

【0097】

また、本実施の形態において、複写を行う場合に考え得る拡張例としては、以下のようなことを挙げることができる。

【0098】

すなわち例えば、子プロセスは親プロセスから複写される情報の内、興味のないものは記憶せずに捨てる例、子プロセスは親プロセスに対して興味のある情報を予め通知することによって、興味のない情報が前記M_COPYのメッセージにて送られてこないようにする例、プロセスの情報を丸ごと一つのM_COPYのメッセージで送るのではなく、複数に分割することによって情報を変更する際の効率を良くする例等が考えられる。

【0099】

次に、プロセスが他のプロセスにアクセスする場合の具体的なアドレス等について以下に説明する。

【0100】

サービスを提供するプロセスは、サービスにアクセスする手段を例えば図13に示すように、サービス識別子、アドレス、ポート番号からなる情報を、テーブルに記憶することによって、共有するようになされている。

【0101】

より具体的に表すと、プロセスは、図14に示すように、サービス識別子とアドレスとポート番号からなるテーブルを記憶することで情報を共有する。すなわち、図14において、サービス識別子として文字列を用い、プロセスのアドレス及びポート番号はTCP/IPのアドレス及びポート番号を使用して記述する。図14では、サービスの内容として例えばWWW (World Wide Web) のサービスとファイルサービスとプリントサービスを例に挙げ、これらのサービス識別子として「WWW-service」と「file-service」と「print-service」が記述され、また、各サービスに対応するTCP/IPのアドレス及びポート番号アドレスとして「192.168.1.1」及び「80」、「192.168.1.1」及び「2049」、「192.168.1.1」及び「515」を一例として挙げている。

【0102】

また、上記の情報を各プロセスで共有した場合、各プロセス内のテーブルは、図15に示すようになる。すなわち、図15において、addrは前述したように複写元のアドレスを示し、さらにこのaddrに対してサービス識別子とアドレスとポート番号が記述されている。サービス識別子、アドレス、ポート番号の内容については、図14と同様である。

【0103】

以上説明したように、本発明実施の形態のネットワークシステムによれば、各プロセスに関するアドレスや機能等の情報が可能な限り一個所（例えば親プロセス）に複写されるので、この一個所（親プロセス）で全体の状況を把握することができる。また、本発明実施の形態のネットワークシステムによれば、各プロセスに関するアドレスや機能等の情報は可能な限り各プロセスに複写可能であるため、各プロセスは常に他のプロセスに関する情報を知ることができる。また、本発明実施の形態のネットワークシステムによれば、各プロセスは利用可能なサービスの一覧を常に知ることができ、それにアクセスすることができる。さらに、本発明実施の形態のネットワークシステムによれば、障害の発生やネットワーク構成の変化に対して、自動的に且つ速やかに対応し、処理を続行することが可能である。

【0104】

【発明の効果】

以上の説明で明らかのように、本発明のネットワークシステム及びネットワーク制御方法においては、第1状態のプロセス部には当該第1状態のプロセス部を記憶している第2状態のプロセス部を記憶し、第2状態のプロセスには第1状態の唯一のプロセス部を記憶し、第1状態のプロセス部の数を1つにすることにより、様々なネットワーク構成に柔軟に対応でき、尚かつネットワーク上で障害が発生しても、各プロセスが所望の処理を継続可能である。

【0105】

また、本発明の信号送受信装置においては、自己に関する情報を記憶している第1状態の唯一の他の機器がネットワークに接続されている時には自己の状態を第2状態として第1状態の他の機器を記憶し、第1状態の自己を記憶している第2状態の他の機器がネットワークに接続されている時には自己を記憶している第2状態の他の機器に関する情報を記憶することにより、様々なネットワーク構成に柔軟に対応でき、尚かつネットワーク上で障害が発生しても、各プロセスが所望の処理を継続可能である。

【図面の簡単な説明】

【図1】

本発明のネットワークシステム及びネットワーク制御方法が適用される一実施の形態としてのネットワークシステムの概略構成を示す構成図である。

【図2】

あるプロセスが親プロセス又は子プロセスの何れかの状態になるまでの流れを示すフローチャートである。

【図3】

2つのグループに属するプロセス間が接続された場合に当該接続されたプロセスが行う処理の流れを示すフローチャートである。

【図4】

自グループ内の子プロセスから他グループの親プロセスを知らせるメッセージを受信した場合の親プロセスにおける処理の流れを示すフローチャートである。

【図5】

親プロセスが子プロセスに変化して新たな親プロセスが存在することになった場合の子プロセスにおける処理の流れを示すフローチャートである。

【図6】

障害の発生により、子プロセスの存在を確認できなくなったときの親プロセスにおける処理の流れを示すフローチャートである。

【図7】

障害の発生により、親プロセスの存在を確認できなくなったときの子プロセスにおける処理の流れを示すフローチャートである。

【図8】

親プロセスの状態である子プロセスの動作の流れを表す図である。

【図9】

子プロセスの状態であるプロセスの動作の流れを表す図である。

【図10】

親プロセス間で新たな親プロセスを決定する際の協議（ネゴシエート）時の動作の流れを表す図である。

【図11】

複写を実現するためにプロセスが記憶するテーブルの基本構成を示す図である。

【図12】

複写を行う場合の動作の流れを表す図である。

【図13】

各プロセスのサービスにアクセスする際ために各プロセスが記憶するテーブルの基本構成を示す図である。

【図14】

サービス識別子とアドレスとポート番号からなるテーブルの一例を示す図である。

【図15】

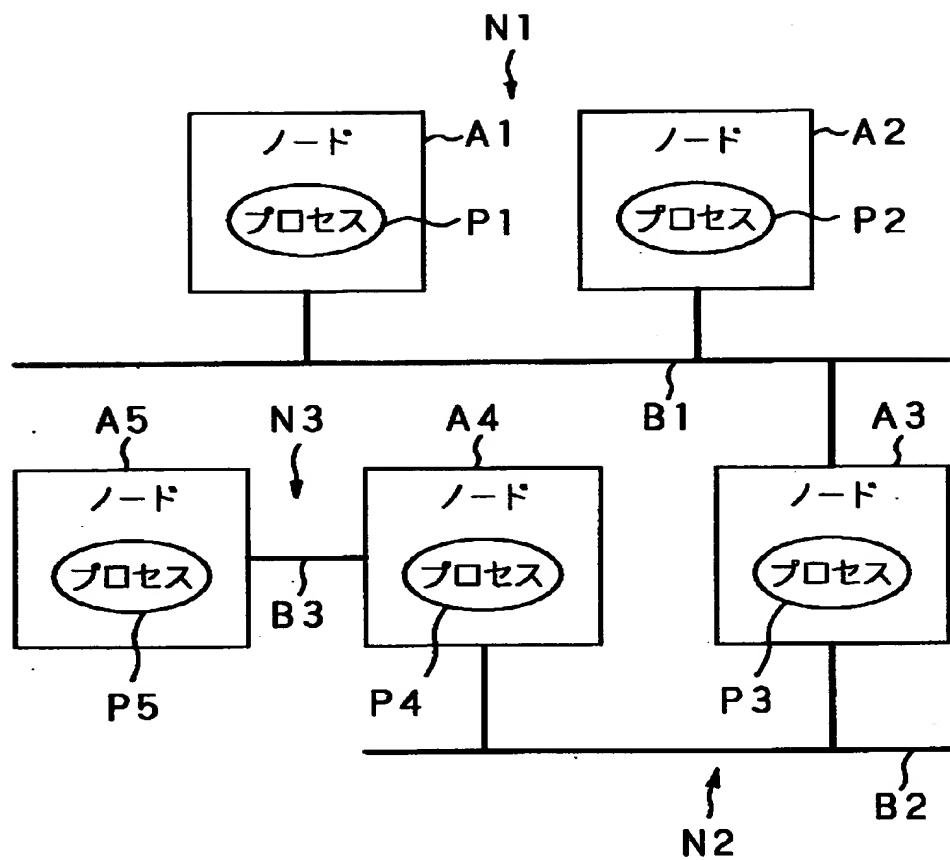
各プロセスで情報を共有した場合のテーブルの一例を示す図である。

【符号の説明】

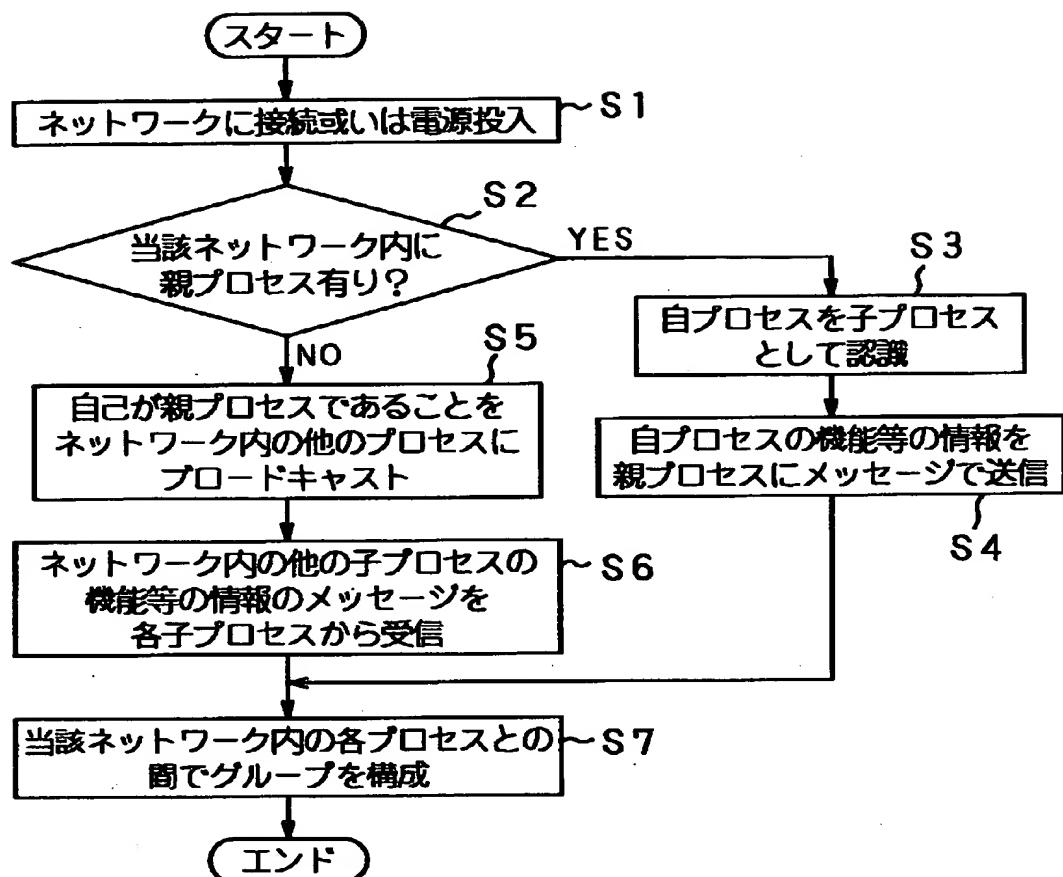
A1, A2, A3, A4, A5 ノード（コンピュータ等）、P1, P2,
P3, P4, P5 プロセス、N1, N2, N3 ネットワーク、B1, B
2, B3 ネットワークセグメント

【書類名】 図面

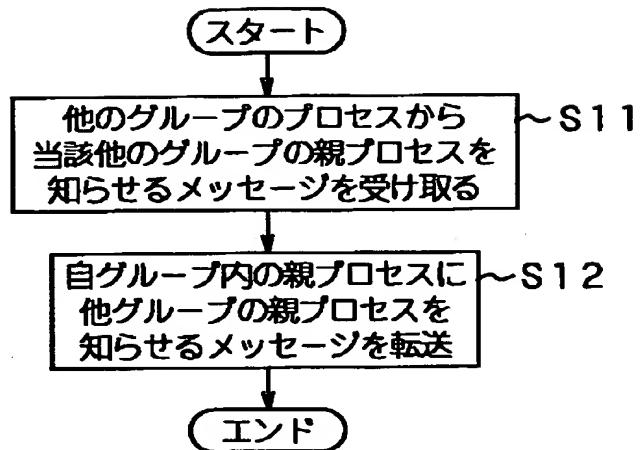
【図1】



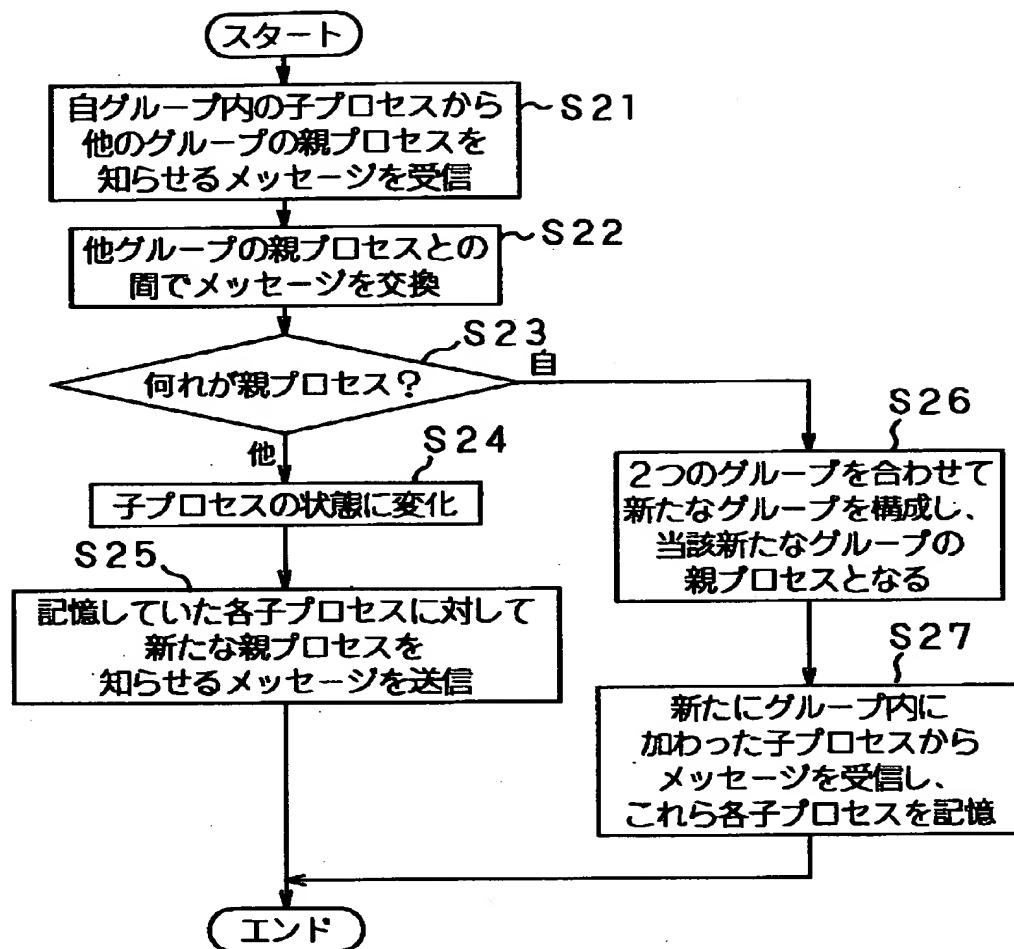
【図2】



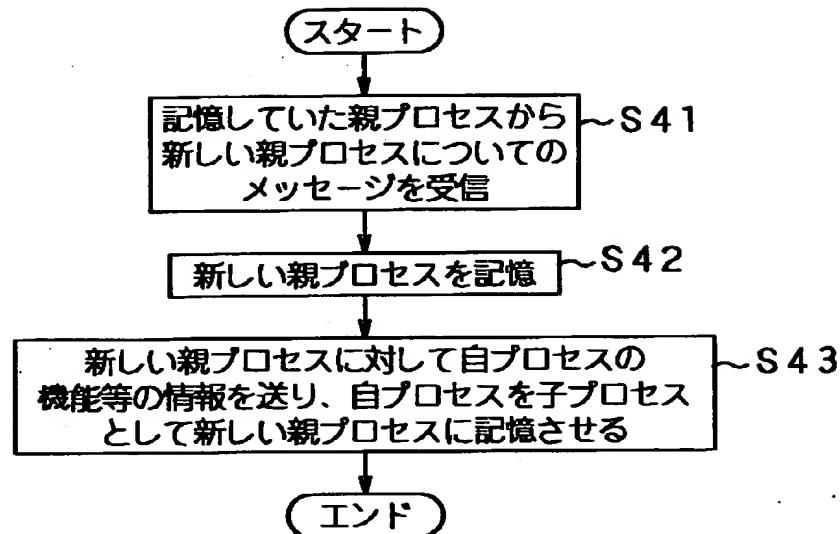
【図3】



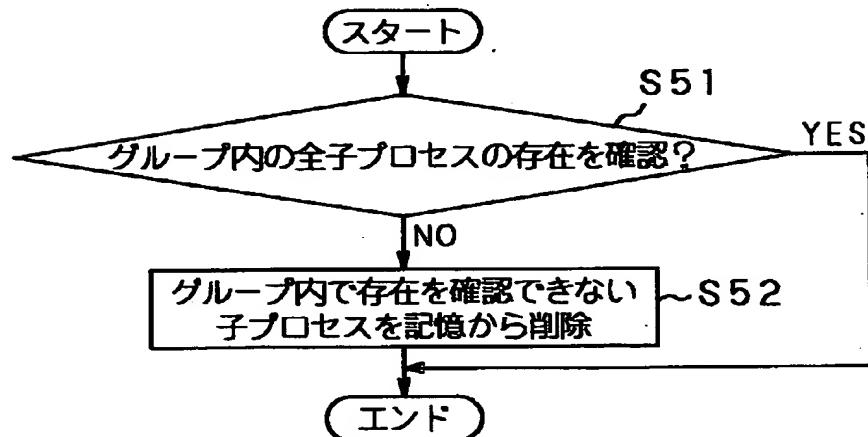
【図4】



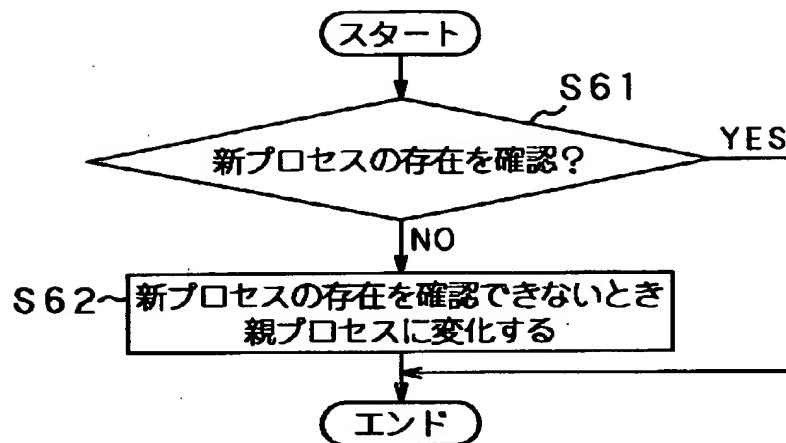
【図5】



【図6】



【図7】



【図8】

```

loop{
  wait

  if(S==S_PARENT){
    if(E.m_type==M_NOTIFY){
      if(E.m_addr!=M){
        assign_P(E.m_addr);
        send(P,M_NEGOTIATE,M);
        trans(S_NEGOTIATING);
      }
    }else
    if(E.m_type==M_FORWARD){
      ignore();
    }else
    if(E.m_type==M_REPORT){
      if(E.m_addr!=M){
        assign_P(E.m_addr);
        send(P,M_NEGOTIATE,M);
        trans(S_NEGOTIATING);
      }
    }else
    if(E.m_type==M_NEGOTIATE){
      assign_P(decide(E.m_addr,M));
      if(P==M){
        send(E.m_addr,M_IAM,M);
        trans(S_NEGOTIATING);
      }
    }else{
      send(E.m_addr,M_YOUARE);
      send(P,M_JOIN,M);
      send_C(M_FORWARD,P);
      empty_C();
      trans(S_CHILD);
    }
  }else
    if(E.m_type==M_IAM){
      ignore();
    }else
      if(E.m_type==M_YOUARE){
        ignore();
      }else
        if(E.m_type==M_BUSY){
          ignore();
        }else
          if(E.m_type==M_JOIN){
            add_C(E.m_addr);
          }else
            if(E.m_type==M_ERROR){
              ignore();
            }else
              if(E.m_type==M_TIMEOUT){
                broadcast(M_NOTIFY,M);
              }
    }
}

```

【図9】

```

if(S==S_NEGOTIATING){
  if(E.m_type==M_NOTIFY){
    ignore();
  }else
    if(E.m_type==M_FORWADR){
  if(E.m_addr==M){
    assign_P(M);
    trans(S_PARENT);
  }else{
    assign_P(E.m_addr);
    send(P,M_NEGOTIATE,M);
  }
  }else
    if(E.m_type==M_REPORT){
      ignore();
    }else
      if(E.m_type==M_NEGOTIATE){
  if(P==E.m_addr && decide(P,M)==-P){
    send(E.m_addr,M_YOUARE);
  }else{
    send(E.m_addr,M_BUSY);
  }
  }else
    if(E.m_type==M_IAM){
      assign_P(E.m_addr);
      send(P,M_JOIN,M);
      send_C(M_FORWARD,P);
      empty_C();
      trans(S_CHILD);
    }else
      if(E.m_type==M_YOUARE){
        assign_P(M);
        trans(S_PARENT);
      }else
        if(E.m_type==M_BUSY){
          send(P,M_NEGOTIATE,M);
        }else
          if(E.m_type==M_JOIN){
            add_C(E.m_addr);
          }else
            if(E.m_type==E_ERROR){
              assign_P(M);
              trans(S_PARENT);
            }else
              if(E.m_type==E_TIMEOUT){
                assign_P(M);
                trans(S_PARENT);
              }
            }else

```

【図10】

【図11】

addr<情報>D[]

【図12】

```

loop{
  自分自身の情報を必要に応じて更新する。
  更新した行のD[]は全ての要素を削除し、Mを追加する。

  if(M_COPY メッセージが到着したか){
    メッセージに応じてテーブルを変更する
    更新した行のD[]は全ての要素を削除し、addrを追加する
  }

  if(M_DELETE メッセージが到着したか){
    テーブルからaddrが一致する行を削除する
  }

  if(自分は親か?){
    for(テーブル内の全ての行について){
      if(addrがC[]に含まれない行を削除する){
        for(C[]の全ての要素Tについて){
          send(T,M_DELETE,addr)
        }
      }
    }
    for(テーブル内の全ての行について){
      for(C[]の要素の内、D[]に含まれない要素Tについて){
        send(T,M_COPY,addr,<情報>)
        D[]にTを追加する
      }
    }
  } else{
    for(テーブル内の全ての行について){
      if(addr==M且つD[]にPが含まれない){
        send(P,M_COPY,M,<情報>)
        D[]にPを追加する
      }
    }
  }
}

```

【図13】

サービス識別子	アドレス	ポート番号
.	.	.
.	.	.
.	.	.

【図14】

サービス識別子	アドレス	ポート番号
WWW-service	192.168.1.1	80
file-service	192.168.1.1	2049
print-service	192.168.1.1	515

【図15】

addr	情報			D[]
	サービス識別子	アドレス	ポート番号	
192.168.1.1	WWW-service	192.168.1.1	80	...
	file-service	192.168.1.1	2049	...
	print-service	192.168.1.1	515	...
192.168.1.2	WWW-service	192.168.1.2	80	...
	print-service	192.168.1.2	515	...
	WWW-service	192.168.1.3	80	...
192.168.1.3	file-service	192.168.1.3	2049	...

【書類名】 要約書

【要約】

【課題】 様々なネットワーク構成に柔軟に対応可能とし、尚かつネットワーク上で障害が発生しても、各プロセスが所望の処理を継続可能とする。

【解決手段】 送り先を指定しないブロードキャストによるメッセージ及び特定のプロセス部を指定したメッセージを相互に送受信し得、親プロセスと子プロセスの何れにも自己の状態を変化し得るプロセス P 1～P 5 を接続してなるネットワークシステムであり、例えばプロセス P 1 が親プロセスで、各プロセス P 2～P 4 が子プロセスであり、これらプロセス P 1～P 4 でグループを構成している場合、各子プロセス P 2～P 4 は親プロセス P 1 を記憶し、親プロセス P 1 は自己 (P 1) を記憶している各子プロセス P 2～P 4 を記憶する。他のグループのプロセス P 5 が接続されたとき、親プロセス P 1 と P 5 との間でメッセージを交換して協議し、何れか一方が親プロセスになり、他方は子プロセスに変化する。

【選択図】 図 1

出願人履歴情報

識別番号 [000002185]

1. 変更年月日 1990年 8月30日

[変更理由] 新規登録

住 所 東京都品川区北品川6丁目7番35号

氏 名 ソニー株式会社

**This Page is Inserted by IFW Indexing and Scanning
Operations and is not part of the Official Record**

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- BLACK BORDERS**
- IMAGE CUT OFF AT TOP, BOTTOM OR SIDES**
- FADED TEXT OR DRAWING**
- BLURRED OR ILLEGIBLE TEXT OR DRAWING**
- SKEWED/SLANTED IMAGES**
- COLOR OR BLACK AND WHITE PHOTOGRAPHS**
- GRAY SCALE DOCUMENTS**
- LINES OR MARKS ON ORIGINAL DOCUMENT**
- REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY**
- OTHER:** _____

IMAGES ARE BEST AVAILABLE COPY.

As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.